BOJ 14889 풀이

스타트와 링크

start

link

Approach

Approach

start

1. 한 쪽 팀을 뽑는다.

➔ 맹목적으로 각각을 돌며 탐색할 것이므로
DFS를 이용하기로 했다.

start

link

2. 각각 능력치를 계산한다.

뽑힌 팀은 `picked[]=true` ,
그렇지 않은 팀은 `picked[]=false` 임을 이용한다.

CODE

변수, main()

```
public static int result=2147483647;
public static int N;
public static int stat[][];
public static boolean picked[];
```

Result:
팀 능력치 계산 후
차이가 최소인 결과를 저장

```
Pick(0,0);

bw.append(result/2+"");
bw.flush();bw.close();br.close();
```

Pick() :
1. 팀을 뽑음
2. 팀을 다 뽑으면 능력치 계산

Pick()

```
public static void Pick(int playerNum, int cnt) {
    if(cnt==N/2) { //한쪽 팀 인원 다 뽑음: 팀 능력치 계산
        int team1=0;
        int team2=0;

        for(int i=0;i<N;i++) {
            if(picked[i]) { //team1의 팀능력치 구함
                for(int j=0;j<N;j++) {
                    if(j==i) continue;
                    if(picked[j]) team1 += stat[i][j] + stat[j][i];
                }
            }else { //team2의 팀능력치 구함
                for(int j=0;j<N;j++) {
                    if(j==i) continue;
                    if(!picked[j]) team2 += stat[i][j] + stat[j][i];
                }
            }
        }

        result=Math.min(result, Math.abs(team1-team2));
    }
```

cnt: 팀원 수.
cnt==n/2 → 팀을 다 뽑았다. → 능력치 계산

Pick()

```
for(int i=playerNum+1;i<N;i++) {
    if(!picked[i]) {
        picked[i]=true;
        Pick(i,cnt+1);
        picked[i]=false;
    }
}
```

cnt < n/2 → 팀을 덜 뽑았다. → DFS방식으로 쭉 뽑는다.